

Documentation MMO NeedAName

Module d'IA



DIETLIN Emmanuel

BLANC Ludovic

CHAMINADE Arthur

POIROUX Martin

Sommaire

[I.A. Fonctionnement général de l'IA](#)

[I.A.1. Système de réputation](#)

[I.A.2. Interactions joueur/PNJ](#)

[I.A.3. Alliances entre PNJ](#)

[I.A.4. Types des PNJ](#)

[I.A.5. Caractères des PNJ](#)

[I.A.6. Noms des PNJ](#)

[I.A.7. Puissance des PNJ](#)

[I.A.8. Boucle de décision des PNJ](#)

[I.B. Fonctionnement détaillé des interactions joueur/PNJ](#)

[I.B.1. Impact des actions du joueur sur la réputation](#)

[I.B.2. Impact de la réputation sur les décisions des PNJ](#)

[I.B.3. Impact des actions du joueur menant à une décision directe du PNJ](#)

[I.B.4. Probabilités d'interaction des PNJ avec les joueurs](#)

[I.C. Changements et ajouts dans le jeu](#)

[I.C.1. Interface utilisateur](#)

[I.D. Notations et remarques](#)

[I.D.1. Documentation](#)

[I.D.2. Code](#)

[II.A. Outils](#)

[II.B. Architecture du projet](#)

[II.B.1. Initialisation des IAs](#)

[II.B.2. Boucle de décision des IAs](#)

[II.C. Génération des builds](#)

[II.C.1. Build pour l'exécution des IAs](#)

[II.C.2. Build pour l'initialisation des IAs](#)

[II.C.3. Ajouter, modifier ou supprimer des IAs](#)

[II.C.4. Exécution des Builds](#)

[II.C.5. Logs](#)

[II.D. Documentation du code](#)

Introduction

NeedAName est un MMO de stratégie et de gestion qui se joue sur navigateur, dans un style space opera. Son principe est assez simple, chaque joueur crée sa propre base et possède des ressources. Il peut par la suite créer des bâtiments avec ces ressources, les améliorer ou même créer des unités. Les interactions entre joueurs sont possibles : on peut échanger des ressources, ou attaquer d'autres joueurs.

L'**idée générale** de l'ajout d'un module d'IA dans ce jeu est de le dynamiser et le complexifier en ajoutant des nouvelles interactions et enjeux pour le joueur. Côté développement, ce module est une partie indépendante du jeu. Dans le jeu en lui-même, l'IA se profile sous forme de **PNJ**. Chaque PNJ possède (au même titre que chaque joueur) une base, avec des ressources et des bâtiments et peut interagir avec les joueurs. Les actions d'un joueur envers un PNJ auront des répercussions sur les représailles du PNJ envers le joueur.

Voici les objectifs concrets de l'apport de ce module au jeu NeedAName : Possibilités d'interactions simples avec les PNJs : échanges, attaques, cadeaux... Les PNJ peuvent aussi proposer des quêtes à chaque joueur, qui rapportent des récompenses si terminées. Ce module permettrait en plus d'embellir la partie PvE du jeu, d'apporter des interactions inédites (quêtes).

I. Présentation de l'IA

I.A. Fonctionnement général de l'IA

I.A.1. Système de réputation

Définition de la réputation

La **réputation** est une valeur qui permet de quantifier la bonté d'une relation entre un joueur et un PNJ : *Plus la valeur de la réputation est **grande**, plus la relation est **bonne**.*

Intervalles de réputation

Soient R_{\min} , R_{\max} deux entiers constants définis par les game designer. La réputation entre un joueur et une IA pourra varier entre R_{\min} et R_{\max} .

R_{\min} correspond à la réputation la plus **négative** possible. Autrement dit, la relation est à son périgée. Inversement, R_{\max} correspond à une relation à son **apogée**.

Soit x un entier tel que $x = (R - R_{\min}) / (R_{\max} - R_{\min})$. x est compris entre 0 et 1, et est tel que $x = 0 \Leftrightarrow R = R_{\min}$, et $x = 1 \Leftrightarrow R = R_{\max}$.
Notons aussi $R_{\text{int}} = R_{\max} - R_{\min}$

$$x = \frac{R - R_{\min}}{R_{\max} - R_{\min}}$$

Pour toute la suite du document, on gardera ces notations : R_{\min} , R_{\max} , R_{int} et x .
Actuellement dans le jeu, $R_{\min} = 0$ et $R_{\max} = 1000$.

Différents stades de la réputation

Voici ci-dessous les différentes réputations avec leur stades associé :

Statut global	Réputation	Stade raffiné	PNJ
GUERRE	$\{x < 0.0625$	Guerre totale	Ennemi juré}
	$0.0625 \leq x < 0.125$	Guerre	En guerre
PAIX	$0.125 \leq x < 0.25$	Hostilité	Hostile
	$\{0.25 \leq x < 0.375$	Méfiance	Méfiant}
	$\{0.375 \leq x < 0.5$	Distance	Distant}
	$0.5 \leq x < 0.625$	Neutralité	Neutre
	$\{0.625 \leq x < 0.75$	Négociations	Négociant}
	$0.75 \leq x < 0.875$	Amitié	Ami
ALLIANCE	$0.875 \leq x < 0.9375$	Alliance	Allié
	$\{0.9375 \leq x$	Allégeance	Allégeant}

Statuts spéciaux

GUERRE et **ALLIANCE** sont des statuts spéciaux temporaires qui peuvent être enclenchés ou supprimés avec d'autres actions qu'une variation de réputation (déclaration de guerre, traité d'alliance, traité de paix, révocation d'alliance...).

I.A.2. Interactions joueur/PNJ

Voici une liste des **interactions possibles** avec les PNJ :

- **Proposition de quête** (PNJ -> Joueur) : plus probable avec une bonne réputation
- **Complétion de quête** (Joueur -> PNJ) : récompenses pour le joueur (réputation, ressources...)

- **Abandon de quête** (Joueur -> PNJ) : perte de réputation pour le joueur
- **Refus de quête** (Joueur -> PNJ) : perte de réputation pour le joueur
- **{Echange commercial** (PNJ <-> Joueur) : gagnant-gagnant, pas (ou peu) de changement significatif de réputation}
- **Attaque** (Joueur -> PNJ) : perte de réputation pour le joueur, et potentiellement une déclaration de guerre (PNJ -> Joueur)
- **Attaque** (PNJ -> Joueur) : si en guerre ou réputation trop faible
- **Cadeau** (Joueur -> PNJ) : gain de réputation pour le joueur
- **Cadeau** (PNJ -> Joueur) : gain de ressources ou de vaisseaux pour le joueur
- **Aide à la défense** (Joueur -> PNJ) : gain de réputation pour le joueur
- **Aide à la défense** (PNJ -> Joueur) : si réputation suffisamment élevée
- **{Demande de troupes** (Joueur -> PNJ) : acceptation si réputation suffisamment élevée, mais perte de réputation}
- **Pillage d'échange** (Joueur -> PNJ) : perte de réputation
- **Pillage d'échange** (PNJ -> Joueur) : si réputation trop faible

I.A.3. Alliances entre PNJ

Chaque PNJ fait partie d'une alliance de PNJ. Ainsi, ils veillent les uns sur les autres. Les actions d'un joueur envers un PNJ faisant partie d'une alliance **influencent aussi sur les PNJ de l'alliance**.

De la même manière, si **deux alliances sont en guerre**, une action envers un PNJ d'une des alliances aura une **influence opposée sur les PNJ de l'autre alliance**.

I.A.4. Types des PNJ

Il existe plusieurs **types** de PNJ :

- **PNJ Delta** : PNJ gérés par le module d'IA, ne répondant pas aux messages, mais interagissant comme expliqué ci-dessus
- **PNJ Gamma** : PNJ gérés par le module d'IA, mais un animateur peut reprendre à tout moment la main dessus
- **PNJ Beta** : PNJ gérés par les animateurs, possibilité de lui envoyer des messages

I.A.5. Caractères des PNJ

Il y a plusieurs **caractères** d'IA :

- **Mercantile** : aime les échanges
- **Pacifiste** : évite les conflits
- **Belliciste** : aime la guerre

En voici une description **plus détaillée** :

Interactions	Échanges	Attaques	Autres
Caractère			
Mercantile	<p>Accepte facilement les transactions, mais négocie des taux de change avantageux pour elle</p> <p>Offre des ressources en échange de services ou d'avantages</p>	<p>Met peu d'argent dans la guerre</p> <p>Peut contre-attaquer</p> <p>Peut se retirer de la guerre pour protéger ses intérêts commerciaux</p> <p>Peut utiliser la menace de sanctions économiques pour obtenir ce qu'elle veut</p> <p>Peut utiliser la menace de sanctions économiques pour obtenir ce qu'elle veut</p>	<p>Coefficient de propagation de réputation plus grand</p> <p>Investit dans les technologies commerciales pour augmenter ses bénéfices (peut se modéliser par le fait qu'elle possède plus de ressources commerciales)</p> <p>Établit des relations commerciales avec d'autres joueurs pour augmenter ses revenus</p>
Pacifiste	<p>Propose échanges et quêtes assez facilement pour éviter l'hostilité</p> <p>Offre des ressources pour éviter des conflits</p>	<p>Attaque seulement en dernier recours</p> <p>Négocie facilement la paix</p> <p>Évite les combats inutiles et préfère négocier plutôt que de recourir à la violence</p>	<p>Favorise des relations d'amitié et d'alliance avec les autres joueurs</p> <p>Peut accepter des concessions pour éviter des conflits</p>
Belliciste	<p>Grande valeur pour les unités</p> <p>N'offre pas de transaction</p>	<p>Peut attaquer à moins d'avoir une amitié avec le joueur</p> <p>Peut attaquer les joueurs sans raison apparente</p>	<p>Plus difficile de devenir ami</p> <p>N'aime pas que tu sois proche d'elle (géographiquement)</p> <p>A une stratégie agressive et imprévisible</p> <p>Peut utiliser la force brute pour</p>

			imposer sa volonté aux autres joueurs Possède une grande armée Peut utiliser des alliances temporaires pour éliminer des ennemis communs, avant de les trahir.
--	--	--	--

En plus de son trait de caractère principal, chaque PNJ possède une **ressource favorite**, auquel elle donnera une valeur plus grande que pour les autres ressources.

I.A.6. Noms des PNJ

Des noms ont été donnés aux PNJ afin d'affiner leur identité.

Voici les noms actuels des PNJ delta en fonction de leur caractère :

Mercantile	Pacifiste	Belliciste
Zorg Seldon Flam Chiktaba Luke Marcheciel	Alice Bob Horus Cobra Nono E.T.	Luke Harlock Caïd Jayce Amogus

I.A.7. Puissance des PNJ

Il existe plusieurs **templates** pour les IA, définissant leur puissance : 1k, 10k, 100k et 1M sont ceux utilisés pour le moment. Leurs bâtiments, ressources, flottes de défense et d'attaque en dépendent (et peuvent aussi dépendre du caractère du PNJ : un PNJ belliciste aura généralement une flotte plus conséquente).

Critère	Base	Flotte d'attaque	Flotte de défense	Flotte de commerce	Tourelles
Template					
1k	Stockage de 5k de chaque ressource	5 dragunes 200	5 dragunes 200	5 dragunes 200	5 dragunes 200
10k	Stockage de 50k de chaque ressource	20 narca 50 dragunes 200	20 narca 50 dragunes 200	20 narca 50 dragunes 200	20 narca 50 dragunes 200
100k	Stockage de 500k de chaque ressource	50 lug 50 impérial f3	50 lug 50 impérial f3	50 lug 50 impérial f3	50 lug 50 impérial f3
1M	Stockage de 5M de chaque ressource	200 narca 500 imperial f3 100 lug	200 narca 500 imperial f3 100 lug	200 narca 500 imperial f3 100 lug	200 narca 500 imperial f3 100 lug

Régénération des flottes

Toutes les 300s (ou les 100s si seulement un pourcentage de la flotte est détruit)

Régénération des ressources

1% de son stockage maximal ressource est régénéré à chaque tour de boucle de décision du PNJ (cf. I.A.8.) .

Chaque PNJ possède un template pour chaque critère (base, flottes...), ainsi qu'un nombre de flottes du template correspondant.

Exemple

Le PNJ Alice possède :

- Template Base **100k**
- **1** * Template FlotteAttaque **100k**
- **1** * Template FlotteDéfense **100k**
- **2** * Template FlotteCommerce **100k**
- **1** * Template Tourelles **100k**

I.A.8. Boucle de décision des PNJ

Chaque PNJ possède une boucle de décision, qui permet de regarder les interactions avec l'ensemble des joueurs un par un. L'intervalle entre deux boucles est de $t_d = 1 \text{ min.}$

I.B. Fonctionnement détaillé des interactions joueur/PNJ

I.B.1. Impact des actions du joueur sur la réputation

Attaques

- **Perte de réputation lors d'une attaque sur un PNJ** : perte fixe, et perte en fonction du pourcentage de la flotte de l'IA et de ses ressources. La deuxième perte peut être exponentielle en fonction des pourcentages.

Soient :

- P_A la valeur de perte fixe de réputation envers un PNJ lorsqu'il est attaqué
- F_{\max} la puissance maximale de la flotte **défensive** du PNJ
- F la flotte du **PNJ** détruite par l'attaque
- S_{\max} la valeur des ressources du PNJ quand le stockage est plein
- S la valeur des ressources du PNJ pillées lors de l'attaque
- d_f une constante représentant le facteur exponentiel de l'énervement en fonction de la totalité de la destruction de la flotte
- d_s la constante équivalente pour le pillage de ressources
- P_v la constante de perte maximale de réputation par énervement (lors d'une destruction totale)

La **perte de réputation P** à cause de l'attaque est de :

$$P = P_A + P_v * [(F/F_{\max})^{d_f} + (S/S_{\max})^{d_s}] / 2$$

Valeurs de référence :

- $P_A = R_{\text{int}} / 40$
- $P_v = R_{\text{int}} / 5$
- $d_f = 1$
- $d_s = 1$

{Pour un PNJ de caractère **belliciste**, cette perte peut être accrue d'un facteur multiplicatif (décision encore sous réflexion)}.

- **Gain de réputation lors d'une attaque du PNJ** : lorsque le PNJ attaque, le montant de réputation gagné est équivalent à celui qu'il aurait été si les rôles étaient inversés.

Ici,

- F_{\max} est la puissance de la flotte **offensive engagée** de la part du PNJ lors de l'attaque

- **F** la flotte du **joueur** détruite par l'attaque
- **Pillage d'un convoi d'un PNJ** : De la même manière qu'une attaque, le pillage d'un convoi engendre une perte de réputation de :

$$P = P_p + P_c * [(F/F_{max})^{df} + (S/S_{max})^{ds}] / 2$$

où P_p est la valeur de perte fixe de réputation lors d'un pillage de convoi, et P_c la constante de perte maximale de réputation par énervement (lors d'une destruction totale) pour un pillage de convoi.

Valeurs de référence :

- $P_p = R_{int} / 15$
- $P_c = R_{int} / 8$

Commerces

- **{Valeurs relatives des ressources pour les PNJ}** : Chaque PNJ met des valeurs sur ses ressources en fonction du nombre de chacune de ses ressources (et de son comportement)
Plus une ressource manque au PNJ plus elle devient précieuse}
- **{Acceptation d'échange par le PNJ}** : Un échange est accepté si le PNJ pense qu'il gagne de la valeur au cours de l'échange : Un PNJ avec peu de cristaux mais beaucoup d'organique acceptera de l'organique contre plus de cristaux. Cette acceptation est plus facile lorsque la relation avec le PNJ est bonne.}
- **{Calcul de valeur d'une ressource dans un échange}** : La valeur des ressources de l'échange est calculée après l'hypothétique échange. **Cas pratique** :
 - Une IA possède 100 cristaux et 100 organique
 - Un joueur lui propose 100 de cristaux contre 100 d'organique
 - La valeur pré-échange est la même mais la valeur post-échange est très inégale. L'IA se retrouverait avec 200 cristaux et 0 d'organique (peu de valeur pour les cristaux et beaucoup pour l'organique alors que le joueur demande autant d'organique que de cristaux)
 - L'IA refuse l'échange}
- **Gain de réputation** : Le joueur gagne de la réputation avec l'IA si jamais l'IA accepte un échange favorable à ses yeux. La valeur du delta favorable pour l'IA en comptant la rareté des ressources post-échange est proportionnelle au gain de réputation.}

- **{Valeur (relative) d'une ressource selon l'IA}** en fonction de la quantité qu'il possède, et ses ressources totales. Soit **V** la valeur totale de ses ressources, **V_A** la valeur de la ressource **A**. La valeur que l'IA accorde à la ressource **A** est

$$V_{AR} = k_{A1} * V_{AB} * e^{-k_{A2} * V_A/V}$$

- **k_{A1}** est une constante de préférence de la ressource **A** de l'IA.
- **k_{A2}** est le coefficient de saturation de la ressource **A** de l'IA.
- **V_{AB}** est la valeur de base de la ressource **A**.

De cette manière, la valeur de la ressource **A** pour une IA donnée varie de **k_{A1}** (sa préférence) lorsqu'elle n'en possède pas, et diminue avec une vitesse dépendante de son coefficient de saturation pour cette même ressource.}

Quêtes

- Dépend de la quête en elle-même (récompenses annoncées dans la quête)
- Le gain de réputation diminue si on met trop de temps à compléter la quête

{Saturation des variations de réputation

- Lorsque beaucoup d'interactions sont faites en un laps de temps, les variations de réputation deviennent plus faibles. Elles reprennent leur cours normal}

{Friction de la réputation

- A chaque **Δt**, la réputation se rapproche de la réputation **neutre**. Soit **k** une constante appartenant à **[0, 1]**. à chaque **Δt**, **R = R - (R - R_N) * k**, où **R_N** est la réputation neutre.}

Impact passif sur la réputation des autres IA

Soient **x_{AB}** entre une IA **A** et une IA **B**, **x_{AJ}** entre **A** et le **joueur**, et **x_{BJ}** entre **B** et le **joueur**. Soit **Δx_{AJ}** la variation de réputation entre **A** et **joueur**.

Soit **M** un multiplicateur passif, **G** le gain de l'interaction faite avec l'IA.

$$\Delta x_{AJ} = G * M$$

Soient **k_A**, **k_B** les coefficients de propagation pour les IA **alliées** où **en guerre**.

On définit deux IA comme **alliées** lorsqu'elles font partie d'une même alliance.

On définit deux IA comme **en guerre** si leurs alliances respectives sont en guerre.

Pour toute IA (hormis le sujet de l'interaction avec le joueur) alliée à l'IA en question (celle avec qui le joueur fait une interaction), le gain de réputation envers l'IA alliée est de **Δx_{AJ} * k_A**.

Pour les IA en guerre avec l'IA interagissant avec le joueur, la perte de réputation pour le joueur vis-à-vis des ces IA est de **Δx_{AJ} * k_B**.

k_A et **k_B** sont des flottants compris entre 0 et 1. On peut choisir arbitrairement les valeurs suivantes pour tester : **k_A = 0.25**, **k_B = 0.1**

I.B.2. Impact de la réputation sur les décisions des PNJ

Déclaration de guerre de l'IA au joueur

Lorsque $x < 0.125$, l'IA déclare la guerre au joueur. {La déclaration dure **30 minutes**. Pendant cette période, la paix ne peut pas être négociée avec l'IA tant que x n'excède pas **0.25**. Après cette période, la paix peut-être de nouveau négociée pour $x \geq 0.125$.}

Proposition d'alliance de l'IA au joueur

Lorsque $x \geq 0.875$, l'IA propose une alliance au joueur. {Le traité d'alliance dure **30 minutes**. Pendant cette période, si la x descend en dessous de **0.75**, l'IA révoque l'alliance avec le joueur. Après cette période, l'IA révoque l'alliance lorsque $x < 0.875$.}

Fréquence des quêtes

La fréquence est de $x * k$ quêtes par heure de jeu, où k est une constante. k augmente avec la réputation. Voir **I.B.4. Probabilités d'interactions** pour plus de détails.

{Échanges avec l'IA

Les échanges sont plus ou moins faciles avec l'IA : pour une réputation équivalente à $x = 0.5$, échange **équitable**. Pour $x \neq 0.5$, le joueur doit payer une somme équivalente à $(4/3 - x) * 3/2$ fois le montant initial de la transaction (équitable). Autrement dit : pour $x = 0$, le joueur doit payer $p = (4/3 - 0) / 3/2 = 2$ fois plus. pour $x = 1$, le joueur paie $(4/3 - 1) * 3/2 = 1/3 * 3/2 = 1/2$: il ne paie que la moitié. Possibilité de gains de réputation avec l'IA lors d'un échange favorable pour elle (voir **Cadeaux envers IA**).}

Cadeau de la part de l'IA envers le joueur

Lorsque la réputation avec une IA est bonne, l'IA peut spontanément offrir des cadeaux.

Valeur du cadeau : le PNJ offre un pourcentage c de chacune de ses ressources. c varie entre 1 et 6 en fonction de x : $c(\%) = 20 * (x - 0.7)$. Pour une IA belliciste, les cadeaux peuvent se faire en flotte plutôt qu'en ressources.

Cadeau de la part d'un joueur envers une IA

On évalue le montant du cadeau. Soit v_i la valeur initiale totale des ressources de l'IA. Le joueur donne le cadeau. L'IA prend les ressources qu'elle peut prendre (et stocker). Soit v_f la valeur finale totale des ressources de l'IA. Soit v_s la valeur des ressources totales de l'IA, si tout son stockage était plein. Le **gain de réputation** pour le joueur est de

$$(v_f - v_i) / (v_f + v_s) * k_g$$

où k_g est une constante de gain de réputation pour les cadeaux.

Prenons $k_g = (R_{\max} - R_{\min}) * GC_{\max}$, où R_{\min} et R_{\max} sont les constantes de réputation minimum et maximum entre un joueur et une IA. Pour résumer, un cadeau représentant

une fraction x de la valeur finale des ressources de l'IA, le joueur verra sa réputation gagner $x * GC_{max}$ de l'amplitude de la réputation totale. Mettons $GC_{max} = 0.25$. Pour un cadeau doublant les ressources de l'IA, le joueur verra sa réputation passer de 0.3 à 0.55.

Protection de la part d'un PNJ

Lorsqu'un PNJ défend un allié, il envoie un pourcentage de sa flotte P_D .

$$P_D = 10 + 2 * C_A + Q_A / 2 + L_A - (2 * D_D + C_D + Q_D / 2)$$

où :

- C_A est le niveau du chantier spatial de l'attaquant
- Q_A est le niveau du QG de l'attaquant
- L_A est le niveau du laboratoire de l'attaquant
- D_D est le niveau de la plateforme de défense du défenseur
- C_D est le niveau du chantier spatial du défenseur
- Q_D est le niveau du QG du défenseur

Cette valeur P_D est capée entre **10** et **100**.

Attaque de la part d'un PNJ

Lorsqu'un PNJ attaque, il commence par **choisir une base** de façon aléatoire et **équiprobable**.

On définit ensuite la puissance défensive d'une base I de la manière suivante :

$$B_I = D_I + C_I + Q_I / 2$$

où :

- D_I est le niveau de la plateforme de défense de la base I
- C_I est le niveau du chantier spatial de la base I
- Q_I est le niveau du QG de la base I
- L_I est le niveau du laboratoire de la base I

On définit les bases suivantes :

- **P** la base principale du défenseur
- **S1** la base secondaire n°1 du défenseur
- **S2** la base secondaire n°2 du défenseur

Le PNJ envoie un pourcentage de sa flotte P_A

$$P_A = 10 + B_P + B_{S1} + B_{S2}$$

La valeur maximale de P_A est capée à **100**.

La réputation augmente légèrement après une attaque, pour éviter les attaques à répétition sur un même joueur.

I.B.3. Impact des actions du joueur menant à une décision directe du PNJ

{Demande de troupes

le PNJ envoie des troupes si le joueurs lui demande selon le niveau de réputation}

Complétion de quêtes

le PNJ donne la récompense annoncée sur la quête au joueur

{Acceptation d'un échange

Non implémenté pour le moment}

I.B.4. Probabilités d'interaction des PNJ avec les joueurs

Chaque joueur possède vis-à-vis d'un PNJ, une probabilité d'interaction qui augmente avec le temps (et ceci pour chaque type d'interaction). Les probabilités d'interaction dépendent aussi de la réputation entre le joueur et le PNJ (Guerre, Hostilité, Neutralité, Amitié, Alliance).

Les probabilités fonctionnent par incrémentation : à chaque tour de boucle de décision du PNJ (t_d), la probabilité d'interaction augmente de I , jusqu'à arriver au seuil S .

Voici ci-dessous les valeurs utilisées en fonction des types d'IA :

Caractère	Interaction	Attaque		Quête		Cadeau	
	Réputation	I	S	I	S	I	S
Mercantile	Guerre	0.03	0.5				
	Hostilité	0.005	0.25				
	Neutralité			0.003	0.3	0.01	0.1
	Amitié			0.02	0.5	0.05	0.5
	Alliance			0.05	1	0.1	0.7
Pacifiste	Guerre	0.03	0.25				
	Hostilité			0.0001	0.1		
	Neutralité			0.002	0.3		
	Amitié			0.01	0.5	0.03	0.3
	Alliance			0.03	1	0.05	0.5
Belliciste	Guerre	0.1	1				
	Hostilité	0.02	0.4				
	Neutralité	0.001	0.1	0.001	0.3		
	Amitié			0.01	0.5	0.01	0.25
	Alliance			0.05	1	0.03	0.4

Soutien d'un joueur de la part d'un PNJ

Lors d'un combat :

- Si l'un des deux combattants (attaquant ou défenseur) est un **allié**, le PNJ aide l'**allié** (dans son attaque ou sa défense) avec une probabilité de **10%**.
- Si l'un des deux combattants (attaquant ou défenseur) est un **ennemi**, le PNJ aide l'**opposant de l'ennemi** (dans son attaque ou sa défense) avec une probabilité de **10%**.

{Probabilités lorsque le joueur ne joue pas

Lorsque le joueur est inactif ou déconnecté, les probabilités d'interaction sont plus faibles.}

Réinitialisation des probabilités

Lorsqu'une action est entreprise par un PNJ, sa probabilité d'interaction pour l'action en question est remise à zéro.

I.C. Changements et ajouts dans le jeu

I.C.1. Interface utilisateur

Vision de la réputation

Chaque joueur peut voir sa jauge de réputation avec un PNJ en allant dans l'onglet **profil**. Il peut aussi voir son statut actuel de réputation avec l'ensemble des PNJ. La jauge comporte des séparations pour les différents stades de réputation. Chaque PNJ a son nom affiché d'une {couleur} correspondant à son caractère : **Mercantile**, **Pacifiste**, **Belliciste**, ainsi que la jauge à droite de son nom.

{Warning lors d'une attaque

Pour éviter d'attaquer un PNJ par erreur, un message de warning s'affiche lorsque l'on souhaite faire une attaque, prévenant le joueur qu'il reste de perdre de la réputation avec le PNJ en question.}

{Vision des ressources du PNJ

Lorsqu'on fait un échange avec un PNJ, il y a la possibilité de voir les ressources qu'elles possèdent.}

{Demande d'alliance ou déclaration de guerre

Réception de notification pour le joueur de la part du PNJ, lorsque ce dernier propose au joueur une alliance, ou lui déclare la guerre.}

{Fin de traité d'alliance ou de guerre

Réception de notification du joueur, pour annoncer la fin du statut spécial avec un PNJ.}

{Vision des alliés et ennemis des PNJ

Il est aussi possible de voir de quelle alliance une IA fait partie en cliquant dessus. On peut aussi voir les relations entre les différentes alliances (**alliées**, neutres ou **ennemies**).}

I.D. Notations et remarques

I.D.1. Documentation

Des **codes couleurs** sont utilisés au cours de cette documentation, afin d'améliorer la **lisibilité** et l'**attractivité** de ce document.

Pour les parties non implémentées, elles sont encadrées par **{ }**.

I.D.2. Code

Dans le code, nous assimilons les caractères des IA de la façon suivante :

- **Mercantile = Trader**
- **Pacifiste = Pacifist**
- **Belliciste = Warmonger**

II. Documentation technique

II.A. Outils

Le module d'IA du MMO NeedAName est développé à l'aide de **Microsoft Visual Studio 2022**, sur Windows. Il est codé en **C#**, et s'appuie sur la version 7.0 du framework **.NET**.

Le projet utilise deux packages disponibles via NuGet :

- **MongoDB.Driver**, qui est la librairie officielle permettant de communiquer avec MongoDB
- **RabbitMQ.Client**, qui est la librairie officielle permettant d'utiliser **RabbitMQ**

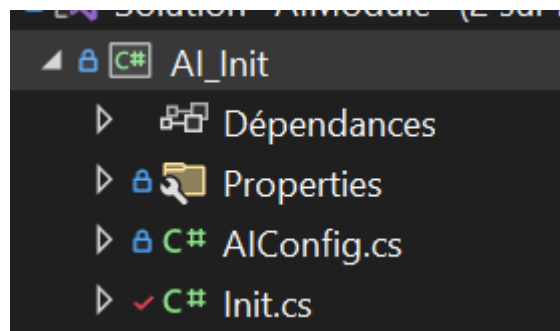
En plus de ces packages, est aussi utilisé le système de sérialisation JSON par défaut de .NET, **System.Text.Json**. Est également utilisé [LINQ](#), qui permet d'effectuer des requêtes sur des sets de données, peu importe leur type.

II.B. Architecture du projet

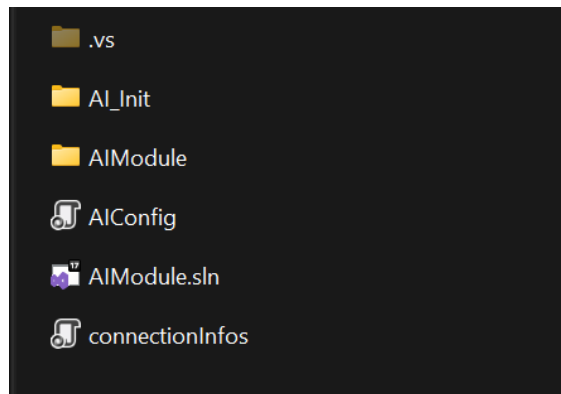
Le projet se compose de deux sous-projets, **AI_Init** et **AIModule**.

II.B.1. Initialisation des IAs

AI_Init est le projet qui va initialiser les IAs ainsi que tous les éléments associés (base, flottes, position sur la planète) dans la base de données. Le fichier qui contient la classe **Main**, qui sera donc exécuté au lancement du programme, est **Init.cs**.



L'application utilise deux fichiers de configuration pour récupérer les informations nécessaires à l'initialisation des IAs, **AIConfig.json** et **connectionInfos.json**.



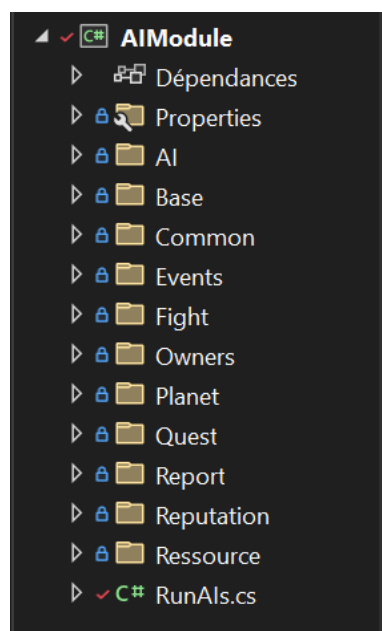
AIConfig.json contient toutes les informations nécessaires à l'initialisation des IAs, notamment le nom, la liste des différents templates, la position, etc...

ConnectionInfos.json contient les informations nécessaires à la connexion à la base de données MongoDB ainsi qu'au service RabbitMQ.

Ces fichiers ne sont pas à l'intérieur du projet, ils se trouvent à la racine du dossier contenant les deux projets. **Il faut penser à les placer dans le dossier de l'exécutable avant de lancer celui-ci.**

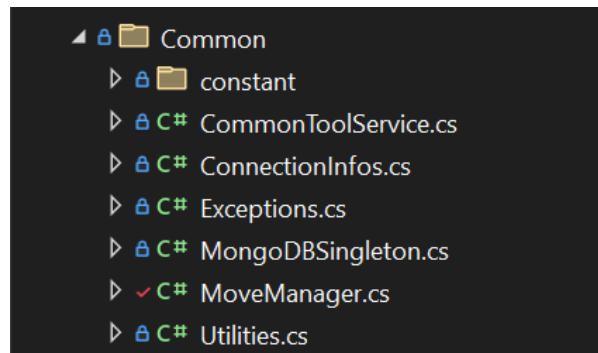
II.B.2. Boucle de décision des IAs

AIModule est le projet qui contient la boucle de décision des IAs. Il représente donc en quelque sorte le cerveau de celles-ci. Les IAs sont récupérées depuis la base de données, puis chacune est lancée sur un thread séparé.



Le projet contient notamment une partie des classes existantes dans les autres modules, qui ont été transcrites en C#. Le programme exécuté au lancement est contenu dans **RunAI.cs**.

Ce projet utilise le fichier **connectionInfos.json** pour récupérer les informations de connexion vers RabbitMQ et MongoDB. Il est important de penser à placer ce fichier dans le dossier de l'exécutable.



Parmi les classes notables, on peut notamment citer la classe **Utilities.cs**, qui contient un ensemble de méthodes pour effectuer diverses tâches et contient également une classe statique avec l'ensemble des constantes utilisées dans la boucle de décision de l'IA.

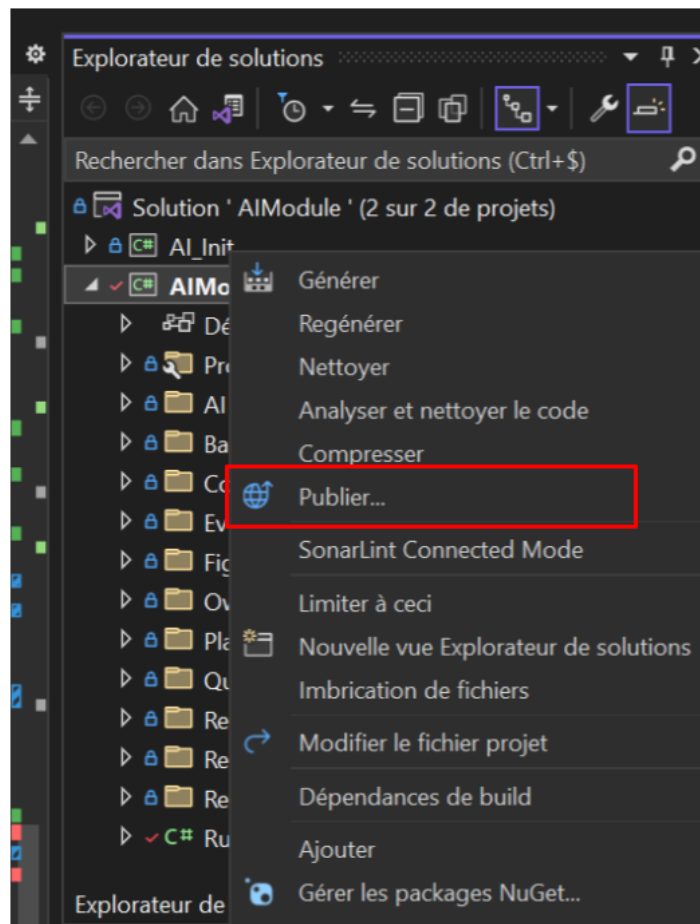
On a également la classe **MongoDBSingleton**, qui contient un point d'accès vers toutes les collections de la base de données MongoDB.

Enfin, la classe **MoveManager** est une classe statique qui offre une méthode pour créer un évènement **FleetMoveEvent**. La méthode est reprise de celle présente dans un autre module mais adaptée pour l'IA spécifiquement.

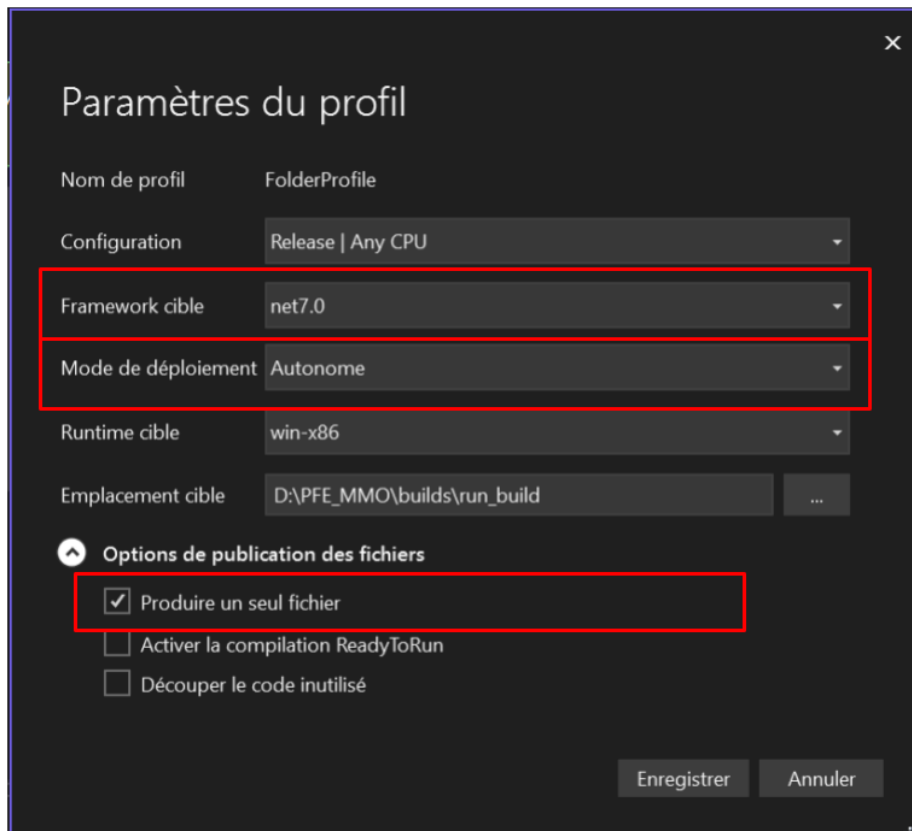
II.C. Génération des builds (avec VS 2022)

II.C.1. Build pour l'exécution des IAs

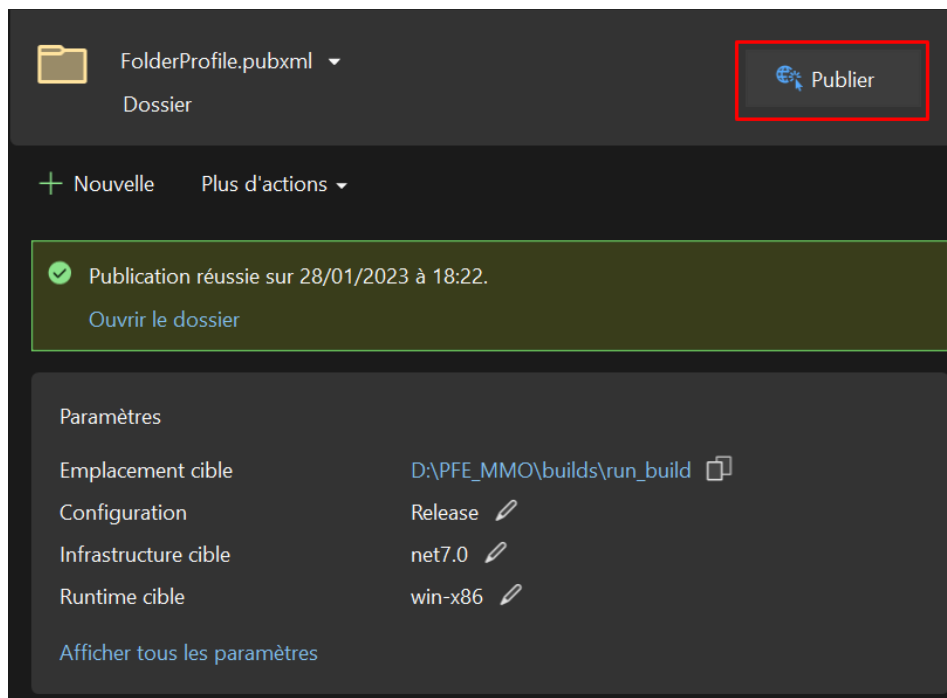
Pour créer un exécutable pour le module AIModule, il faut tout d'abord aller dans l'arborescence du projet, et faire un **clic droit** sur le nom du module (**AIModule**). Il faut ensuite sélectionner l'option "publier" dans le menu contextuel.



Dans l'onglet qui s'ouvre, il faut cliquer sur "afficher tous les paramètres", puis configurer les différentes options selon le build que l'on veut générer. Parmi les paramètres importants, il faut placer **Framework cible** à **net7.0**, **Mode de déploiement** à **Autonome**, et cocher **Produire un seul fichier** dans **Options de publication des fichiers**. Le reste des informations est à adapter selon la plateforme ciblée et le dossier souhaité pour le build.



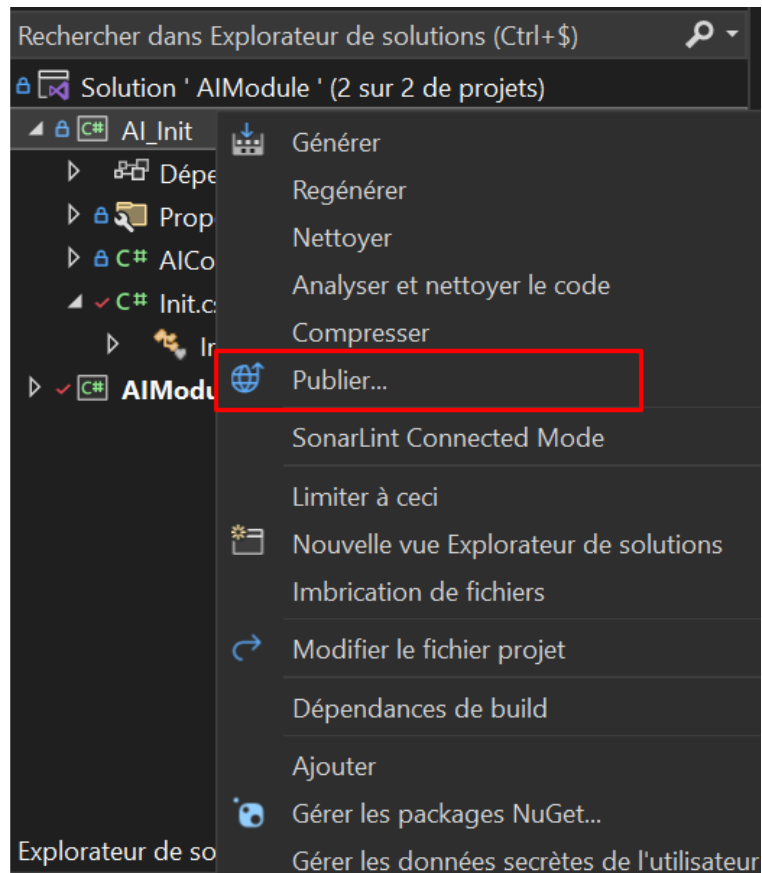
Une fois cette étape effectuée, il faut sélectionner **Enregistrer**, puis enfin cliquer sur **Publier**. S'il n'y a pas de problèmes, le message de publication réussie devrait apparaître.



II.C.2. Build pour l'initialisation des IAs

Pour créer un exécutable pour le module AI_Init, il suffit de reproduire les mêmes étapes que pour le build de AIModule.

Il est important de noter qu'il faut **d'abord** faire le build de AIModule avant de faire le build de AI_Init.



II.C.3. Ajouter, modifier ou supprimer des IAs

La configuration des IAs se trouve dans le fichier **AIConfig.json**. Pour ajouter, modifier ou supprimer des IAs, il faut donc modifier le fichier précédemment cité. Le fichier se présente tel que suit:

```
[
  {
    "name": "Alice",
    "description": "AI Delta",
    "personality": "Pacifist",
    "baseTemplate": "NPC_TEMPLATE_10k",
    "fleetTemplates": {
      "attackFleet": "NPC_TEMPLATE_10k",
      "deliveryFleet": "NPC_TEMPLATE_10k",
      "defenseFleet": "NPC_TEMPLATE_10k",
      "turretFleet": "NPC_TEMPLATE_10k"
    },
    "numberOfFleets": {
      "attackFleet": 1,
      "deliveryFleet": 1,
      "defenseFleet": 2,
      "turretFleet": 1
    },
    "allianceName": "Imperium",
    "allianceId": "Imperium",
    "enemyAllianceIds": [ "Fondation" ],
    "allyAllianceIds": [],
    "x_coord": 74,
    "y_coord": 81,
    "planetName": "TestLand",
    "type": "AIDELTA"
  },
]
```

name : nom de l'IA

description : description de l'IA

personality : **Pacifist** ou **Warmonger** ou **Trader**

baseTemplate : nom du template de la base, parmi les bases de type "NPC_TEMPLATE"

fleetTemplates : dictionnaire avec comme clé **attackFleet**, **deliveryFleet**, **defenseFleet**, **turretFleet**, et comme valeur le nom du template de flotte, parmi les flottes de type "NPC_TEMPLATE"

numberOfFleets: dictionnaire avec comme clé **attackFleet**, **deliveryFleet**, **defenseFleet**, **turretFleet** et comme valeur le nombre de flottes de ce type

allianceName: nom de l'alliance de l'IA

allianceId: id de l'alliance de l'IA

enemyAllianceIds: liste des ids des alliances IA qui sont ennemies de l'IA

allyAlliancesIds: liste des ids des alliances IA qui sont alliées à l'IA

x_coord: position x de la base IA

y_coord: position y de la base IA

planetName: nom de la planète où se trouve la base IA

type: **AlGAMMA** ou **AIDelta**

Pour ajouter une IA, il suffit d'ajouter un objet à la suite des autres dans la liste des objets (la liste est définie par les crochets ouvrants et fermants **[]**).

II.C.4. Exécution des builds

Pour générer ou régénérer les IAs en base de données, il faut exécuter le build **AI_Init** qui se trouve dans le dossier correspondant au build du module **AI_Init**. Il faut bien penser à ajouter les fichiers **connectionInfos.json** et **AIConfig.json** au même niveau que l'exécutable avant de lancer celui-ci.









Il est important de noter qu'il n'y a **pas besoin** de lancer l'exécutable **AIModule** qui se trouve dans le même dossier.

Pour exécuter le comportement des IAs, il faut lancer l'exécutable **AIModule** qui se trouve dans le dossier correspondant au build du module **AIModule**. Il faut bien penser à ajouter le fichier **connectionInfos.json** au même niveau que l'exécutable avant de lancer celui-ci.

Il est important de noter que cet exécutable va générer un dossier **logs**, dans lequel vont se trouver les **logs du comportement** des IAs ainsi que des **logs d'erreur**. Ces logs sont présentés dans la partie suivante.

II.C.5. Logs

L'exécutable **AIModule** va générer un dossier **logs**, dans lequel vont être générés deux types de fichiers de log : **AILogs.txt** et **Error_logs.txt**.

 Allogs	26/01/2023 21:31	Fichier TXT
 Allogs_2023-01-26_21-15-23 - Copie (2).gz	26/01/2023 21:30	Fichier GZ
 Allogs_2023-01-26_21-15-23 - Copie (3).gz	26/01/2023 21:30	Fichier GZ
 Allogs_2023-01-26_21-15-23 - Copie (4).gz	26/01/2023 21:30	Fichier GZ
 Allogs_2023-01-26_21-15-23 - Copie.gz	26/01/2023 21:30	Fichier GZ
 Error_logs	26/01/2023 21:31	Fichier TXT
 Error_logs_2023-01-26_21-15-23.gz	26/01/2023 21:30	Fichier GZ
 Error_logs_2023-01-26_21-30-38.gz	26/01/2023 21:31	Fichier GZ

Allogs contient un résumé de toutes les **EventResults** qui sont **traités** par l'IA, ainsi que toutes les **actions** qui sont effectuées par l'IA.

Error_logs contient toutes les exceptions qui sont levées lors de l'exécution afin de permettre de déboguer plus facilement par la suite.

Si la taille des fichiers de logs dépasse 100 Mo ou bien lors du lancement de l'exécutable, le fichier est archivé et un nouveau est créé. Seules les 10 archives les plus récentes sont conservées dans le dossier. Les archives sont au format **.gz**.

II.D. Documentation du code

La documentation au format .html peut être retrouvée dans le dossier **documentation** à la racine du repository git. Pour la lire, il suffit de double-cliquer sur le raccourci **Documentation**.

Elle à été générée à l'aide de l'outil **Doxygen**, dont le site se trouve à [cette adresse](#).

Documentation Module IA

Main Page

Packages

Classes

Files

Documentation Module IA

Packages

Package List

AI_Init

AIModule

AI

AI

AI_Delta

AI_Gamma

Bases

Common

Events

Flight

Owners

Planet

Quest

Report

Reputation

Ressource

Package Members

Classes

Files

AIModule.AI.AI Class Reference

Public Member Functions | Protected Member Functions | Properties | List of all members

Inheritance diagram for AIModule.AI.AI:

```

graph BT
    AIModule_Owners_Owner[AIModule.Owners.Owner] --> AIModule_AI_AI[AIModule.AI.AI]
    AIModule_AI_AI --> AIModule_AI_AI_Delta[AIModule.AI.AI.Delta]
    AIModule_AI_AI --> AIModule_AI_AI_Gamma[AIModule.AI.AI.Gamma]

```

Public Member Functions

virtual void runAI (ConnectionFactory factory)

Méthode permettant d'exécuter le comportement de l'IA.

void ChooseAndExecuteAction (Random rand, string player, IModel channel, IBasicProperties properties, float reputation, Base mainBase)

Pour un joueur donné, regarde si peut effectuer une actions avec lui ou non. Met ensuite à jour la liste des joueurs à considérer et met à jour les flottes de l'IA.

void updateOrSetPlayerReputation (string playerId, float oldReputation, float newReputation)

Met à jour la réputation du joueur en BDD ainsi que la liste des alliés et ennemis de l'IA.

Protected Member Functions

AI ()

Protected Member Functions inherited from AIModule.Owners.Owner

Properties

string personality[get, set]

Personnalité de l'IA. Types de personnalité définis dans la classe statique Personnalité dans le fichier AI.cs

AIModule AI AI

Generated by doxygen 1.9.6

Documentation Module IA

Main Page

Packages

Classes

Files

Documentation Module IA

Packages

Package List

AI_Init

AIModule

AI

Bases

Common

constant

ConnectionInfos

InvalidBodyException

MongoDBSingleton

MongoInfo

MoveManager

RabbitMQInfo

ResourceNotFoundException

TemplateElement

InteractionInformation

InteractionType

PlayerElement

ReputationStatus

Events

Flight

Owners

Planet

Quest

Report

Reputation

Ressource

Package Members

AIModule.Common.MongoDBSingleton Class Reference

Static Public Member Functions | Properties | List of all members

Static Public Member Functions

static MongoDBSingleton Instance ()

Properties

IMongoDatabase database[get]

IMongoCollection< Base > basesCollection[get]

IMongoCollection< Fleet > fleetsCollection[get]

IMongoCollection< FleetMoveEvent > moveEventsCollection[get]

IMongoCollection< User > usersCollection[get]

IMongoCollection< RessourceDetail > ressourcesDetailCollection[get]

IMongoCollection< ReputationObject > reputationCollection[get]

IMongoCollection< Event > eventsCollections[get]

IMongoCollection< QuestTemplate > questTemplatesCollection[get]

IMongoCollection< WaitingQuest > waitingQuestsCollection[get]

IMongoCollection< Alliance > alliancesCollection[get]

IMongoCollection< AI_Delta > aiDeltaCollection[get]

IMongoCollection< AI_Gamma > aiGammaCollection[get]

IMongoCollection< BuildingRef > buildingsCollection[get]

string mongoAddress[get, set]

string mongoName[get, set]

AIModule Common MongoDBSingleton

Generated by doxygen 1.9.6

26

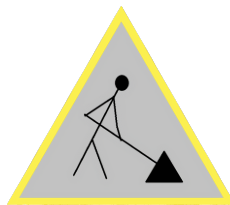
Conclusion

Suggestions et ajouts

Certains points n'ont pas encore pu être implémentés. Voici un **code coloré** pour les repérer dans la documentation : {chose non implémentée}.

En plus de cela, voici ci-dessous une liste d'autres **suggestions d'ajouts**, ou idées qui peuvent être intéressantes à implémenter :

- Avoir plusieurs types de flottes pour chaque IA (pourquoi pas) => notamment une flotte de transport de ressources, qui soit avec bcp de stockage mais également des vaisseaux



Suggestions de nom pour le jeu

- Galactic Empires
- Space Station Siege
- Cosmic Conquest
- Space Base Builders
- Galaxy Wars
- Astro Empire
- Nebula Realms
- Interstellar Siege
- Solar System Strategy
- Orbit Empires
- Stellar Showdown